Heurística Construtiva
Luís Felipe Ramos Ferreira

Como heurística construtiva foram desenvolvidos tanto o algoritmo *twice around the tree* quanto o algoritmo de *Christofides*. No entanto, para simplificar, abordarei nesse relatório apenas o primeiro. O código fonte pode ser encontrado neste repositório, em particular o código do *twice around the tree* pode ser encontrado no arquivo `algorithms.py`.

O algoritmo *twice around the tree* é um algoritmo construtivo e aproximativo para o problema do caixeiro viajante simétrico e que respeita a desigualdade triangular. O algoritmo é 2-aproximativo, ou seja, encontra um caminho do caixeiro que tem comprimento no máximo duas vezes igual ao caminho ótimo. O primeiro passo do algoritmo é a construção da árvore geradora mínima do grafo original,o que pode ser feito de forma eficiente com algoritmos como o de Kruskal. A partir da árvore geradora mínima, escolhe-se um nó qualquer e se executa uma busca em profundidade na árvore. A order dos vértices dessa busca em pré ordem, concatenado com o vértice original, é o caminho final encontrado para o caixeiro.

A tabela abaixo mostra o tempo médio requerido por cada instância quando executadas 20 vezes cada uma. A tabela também mostra o custo do caminho do caixeiro encontrado pelo algoritmo.

instance	path_weight	time(s)
kroD100.tsp	27112	0.005099
lin105.tsp	19495	0.006008
kroB150.tsp	36150	0.012731
berlin52.tsp	10114	0.001455
kroA200.tsp	40028	0.023495
rat195.tsp	3234	0.022293
pr152.tsp	87995	0.012656
st70.tsp	888	0.002403
pr144.tsp	80599	0.011018
pr124.tsp	74139	0.008042
pr136.tsp	151904	0.010339
kroB200.tsp	40703	0.022531
kroB100.tsp	25885	0.006176
kroA150.tsp	35119	0.012949
kroA100.tsp	27210	0.005451
pr107.tsp	54237	0.006193
kroE100.tsp	29965	0.005831
rat99.tsp	1693	0.004874
kroC100.tsp	27968	0.005468
pr76.tsp	145336	0.002865